

게임프로그래밍

# 게임 프로그래밍

## Unity +



유니티를 활용한 게임 프로그래밍

2021763022 김정원  
2023 - 12 - 13

# 목차

## 기본 코드 소개

업그레이드 전 기존의 코드를  
소개하고 개발 방향성 잡기

## UPGRADE

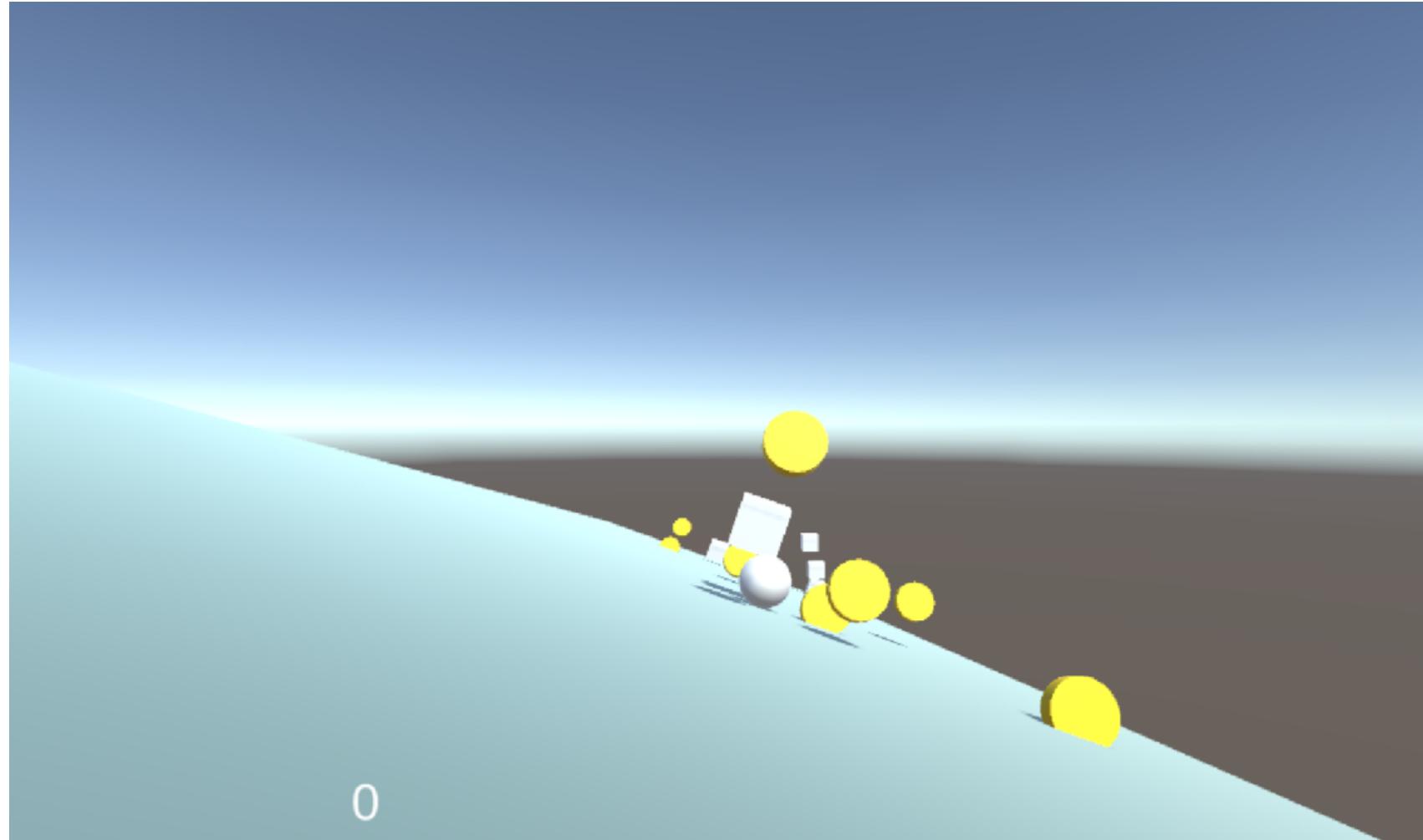
추가된 기능과 구현 방법에 대해  
자세히 알아보기

## 완성작 시연

완성작을 직접 실행해 본 후  
소감 발표하기

# 기본 코드 리뷰

---



게임프로그래밍03(C#Script입문01)

플레이어 이동, 아이템 획득, 게임 재시작, 장애물 등의  
다양한 구성요소가 있어 유니티를 입문하고  
기본 기능을 익히는데 탄탄한 기반이 됨

"수업 내용 복습 및 적극적인 에셋 활용을 목표"

# 주요 업그레이포인트

키워드를 중심으로



game manager의  
재구성

게임 내 맵 재구성  
인트로 아웃트로 구성

게임성

Game  
Manager

font 및 오디오  
기능 추가

가시성

# Get The Ball Rolling

목표지점까지 공을 움직여 목표지점까지 도달하는 게임



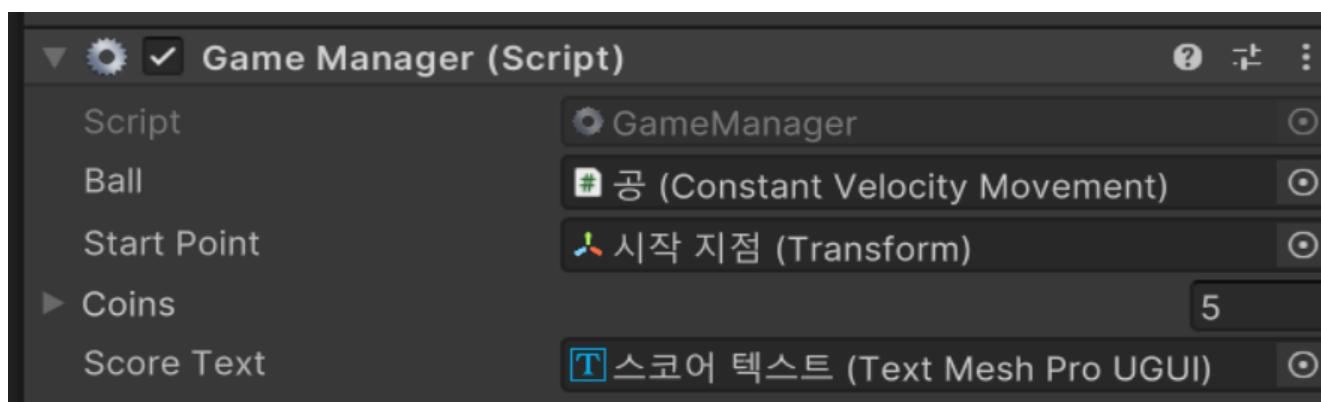
# GAMEMANAGER

## SerializeField

유니티 SerializeField 는 스크립트에서 private 필드를 직렬화하기 위해 사용합니다.

직렬화는 개체의 상태를 나중에 저장, 전송 또는 재구성할 수 있는 형식으로 변환하는 프로세스입니다. 유니티에서 직렬화는 게임 상태를 저장 및 로드하거나 에디터와 런타임 간에 데이터를 전송하는데 사용됩니다.

외부스크립트에서 수정하지 못하며 인스펙터에서는 접근이 가능합니다

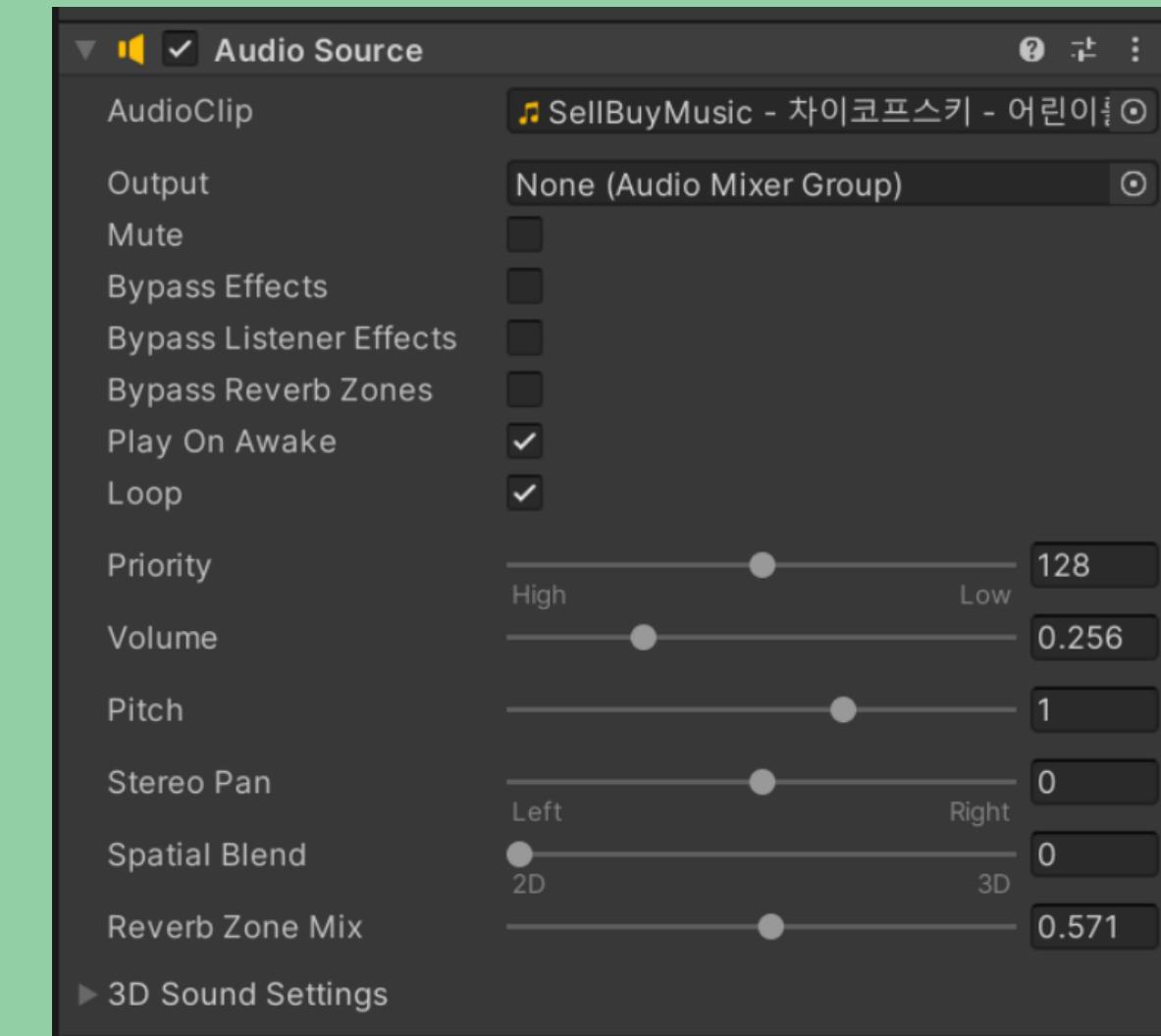


```
public class GameManager : MonoBehaviour{  
  
    [SerializeField] private ConstantVelocityMovement _ball;  
    [SerializeField] private Transform _startPoint;  
    [SerializeField] private List<GameObject> _coins;  
    [SerializeField] private TextMeshProUGUI _scoreText;  
  
    private int _score = 0;  
    private AudioSource[] arrayAudio;  
  
    void Awake(){ StartGame(); }  
  
    void Start(){  
        arrayAudio = GameObject.Find("GameManager").GetComponents< AudioSource>();  
    }  
  
    public void IncreaseScore(int score){  
        _score = _score + score;  
        DisplayScore();  
  
        if (arrayAudio != null && arrayAudio.Length > 1){  
            arrayAudio[1].Play(); }  
    }  
  
    public void ResetScore(){  
        _score = 0; DisplayScore(); }  
  
    private void DisplayScore(){  
        _scoreText.text = _score.ToString(); }  
  
    public void RestartGame(){ StartGame(); }  
  
    public void StartGame(){  
        _ball.Teleport(_startPoint.position);  
  
        foreach (GameObject coin in _coins){  
            coin.SetActive(true);  
        }  
  
        ResetScore(); }  
  
    public void EndGame(){}
}
```

# AUDIO SOURCE

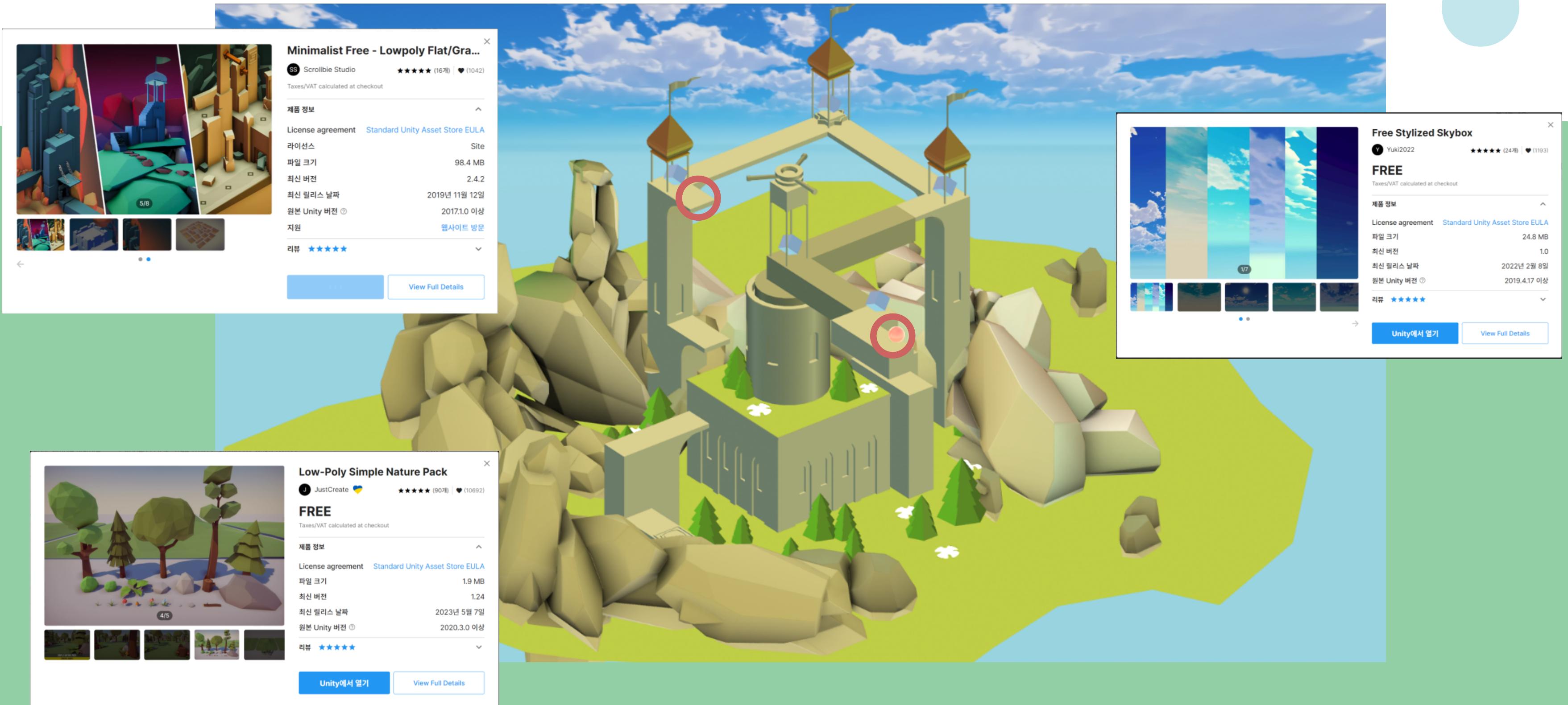
## 배경음악과 효과음

```
audioSource.Play(); //재생  
audioSource.Stop(); //정지  
audioSource.Pause(); //일시정지  
audioSource.UnPause(); //일시정지 해제  
audioSource.playOnAwake = true; //씬 시작시  
바로 재생  
audioSource.loop = true; //반복 재생  
audioSource.mute = true; //음소거  
audioSource.volume = 1.0f; //볼륨 (0.0 ~ 1.0f)  
audioSource.PlayOneShot(audioClip, 1.0f); //  
특정 클립 한번 만 재생  
audioSource.clip = audioClip; //오디오 클립 교체  
  
if (audioSource.isPlaying) Debug.Log("오디오  
재생중입니다.");  
//오디오 재생 여부 확인
```



```
private AudioSource[] arrayAudio;  
  
void Awake(){ StartGame(); }  
  
void Start(){  
    arrayAudio = GameObject.Find("GameManager").GetComponents<AudioSource>();  
}  
  
public void IncreaseScore(int score){  
    _score = _score + score;  
    DisplayScore();  
  
    if (arrayAudio != null && arrayAudio.Length > 1){  
        arrayAudio[1].Play(); } }
```

# 스테이지 구성요소

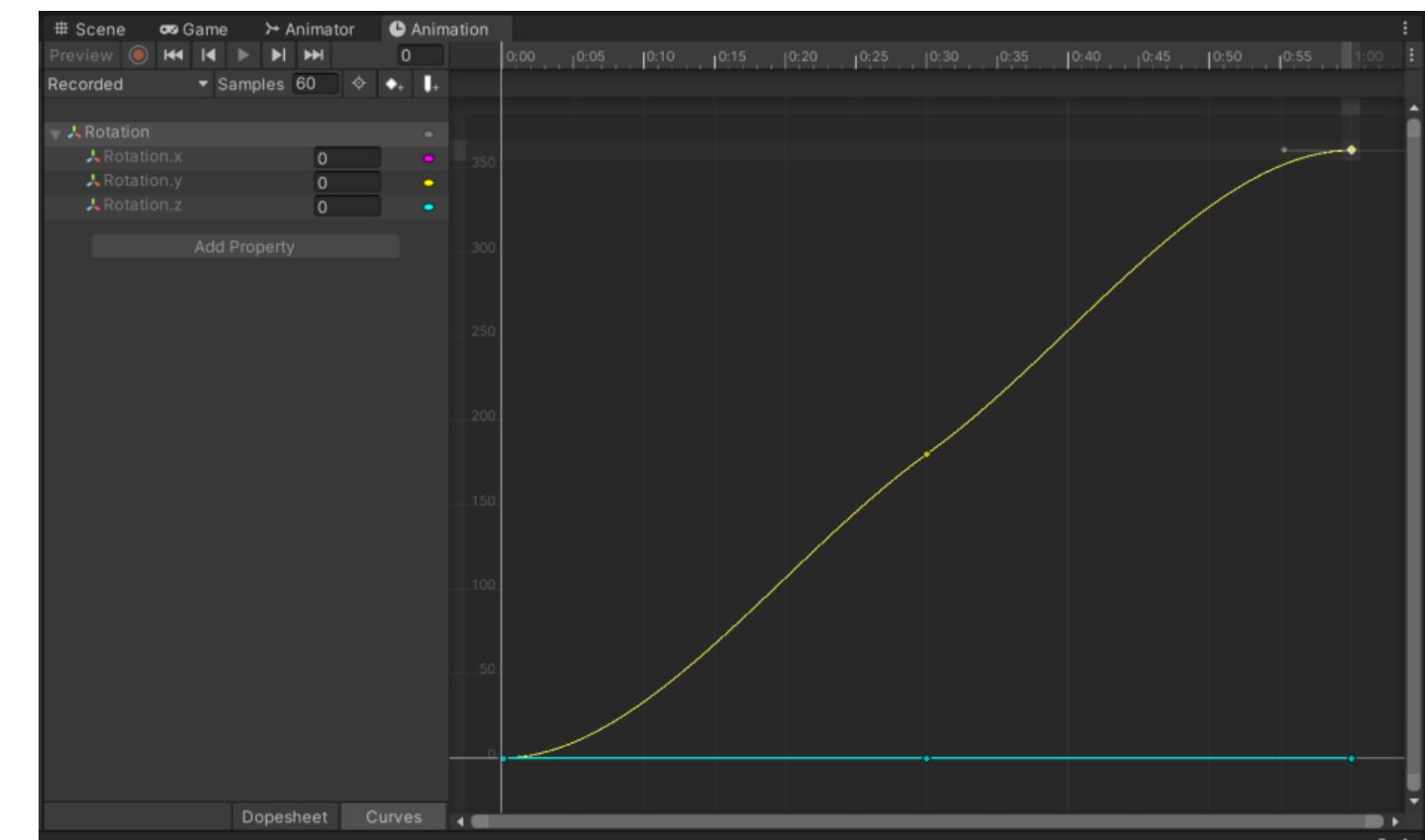
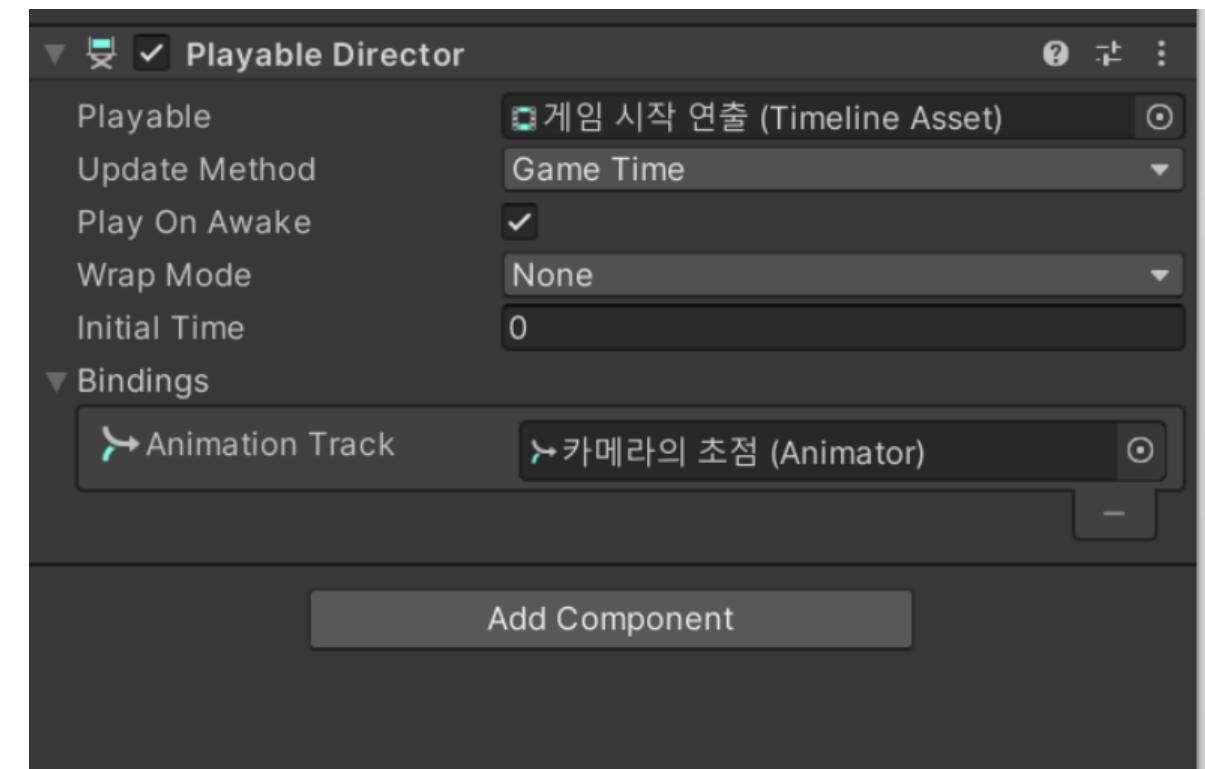


# 스테이지 연출



Animation Clip에서 모든 애니메이션화 프로퍼티에는 Animation Curve가 있어서 Animation Clip에서 그 프로퍼티의 시간에 따른 변화를 조절할 수 있습니다.

Animation View의 프로퍼티 리스트 영역(왼쪽)에는 현재 애니메이션화 프로퍼티가 모두 표시되어 있습니다. 애니메이션 뷰의 도프시트 모드에서 각 프로퍼티의 애니메이션화 값은 선형 트랙으로 보이나, 커브 모드에선 프로퍼티의 변화값을 그래프의 선으로 시각화해서 볼 수 있습니다. 어떤 모드를 사용하든지 커브는 있으며, 도프시트는 키프레임이 발생할 때의 데이터를 단순화해서 보여줄 뿐입니다.



# INTRO

**Image**

- Source Image: start
- Color: None (Material)
- Material: None (Material)
- Raycast Target: ✓
- Raycast Padding: ▶
- Maskable: ✓
- Image Type: Simple
- Use Sprite Mesh: [checkbox]
- Preserve Aspect: [checkbox]

**Button**

- Interactable: ✓
- Transition: Color Tint
- Target Graphic: Button (Legacy) (Image)
- Normal Color: [color swatch]
- Highlighted Color: [color swatch]
- Pressed Color: [color swatch]
- Selected Color: [color swatch]
- Disabled Color: [color swatch]
- Color Multiplier: 1
- Fade Duration: 0.1
- Navigation: Automatic
- Visualize

**On Click ()**

- Runtime Only
- GameObject.SetActive
- GameStart



**Rect Transform**

Pos X	Pos Y	Pos Z
-805.5801	-323.73	0
Width	Height	
100	100	

**Anchors**

Min X	0.5	Y 0.5
Max X	0.5	Y 0.5
Pivot X	0.5	Y 0.5

**Rotation**

X 0	Y 0	Z 0
-----	-----	-----

**Scale**

X 9	Y 5.5	Z 1
-----	-------	-----

**Canvas Renderer**

- Cull Transparent Mesh: ✓

**Image**

- Source Image: header
- Color: None (Material)
- Material: None (Material)
- Raycast Target: ✓
- Raycast Padding: ▶
- Maskable: ✓
- Image Type: Simple
- Use Sprite Mesh: [checkbox]
- Preserve Aspect: [checkbox]

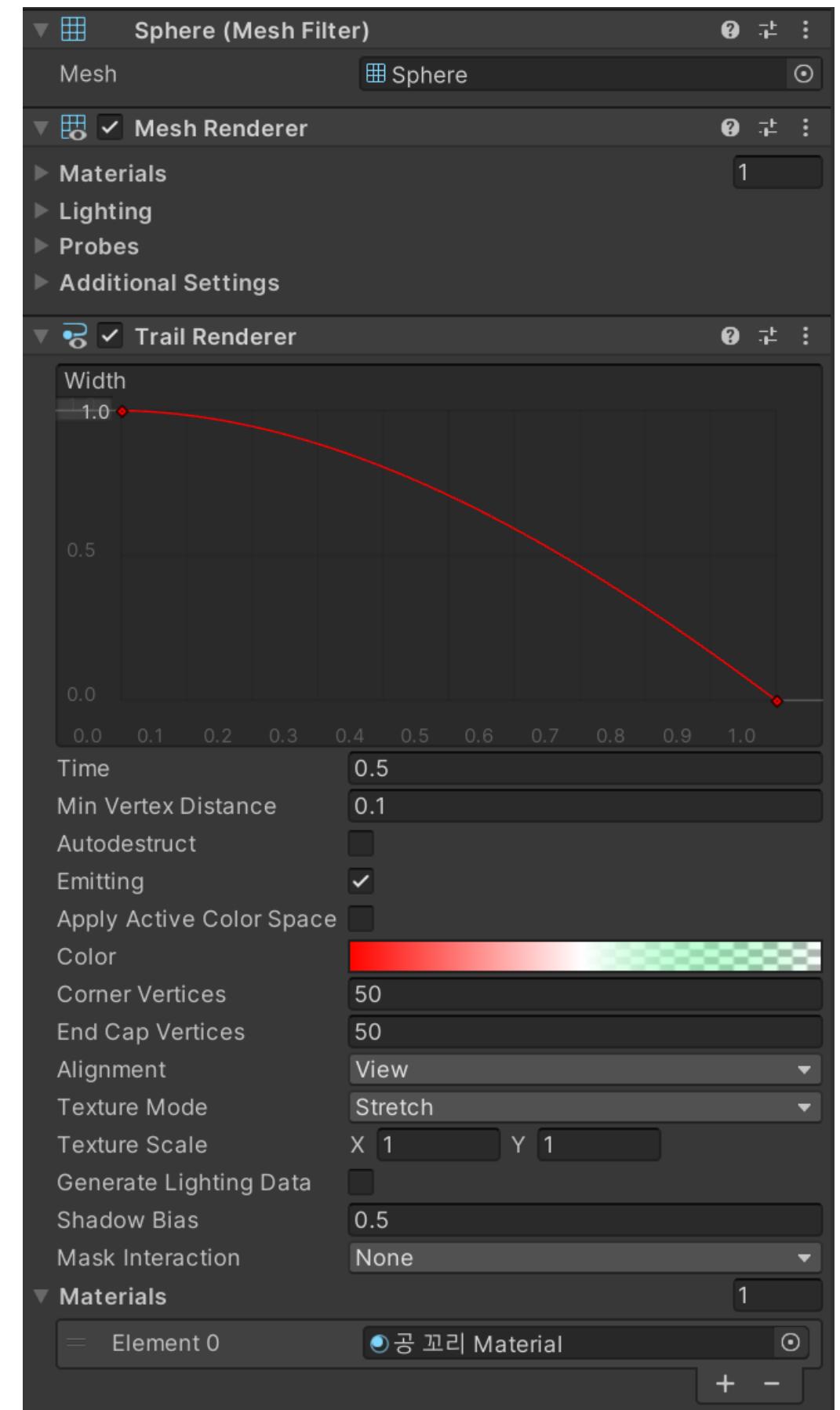
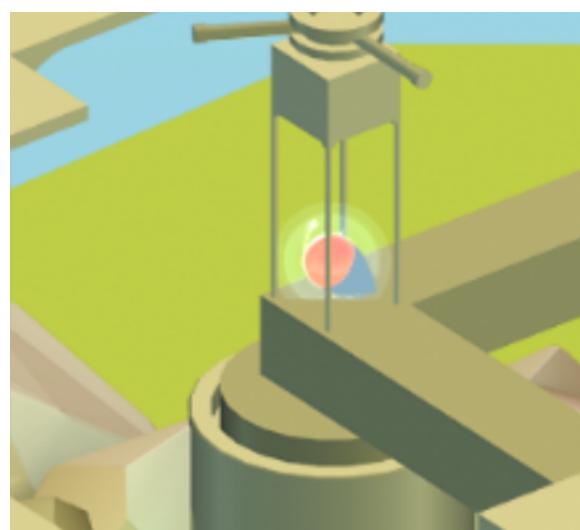
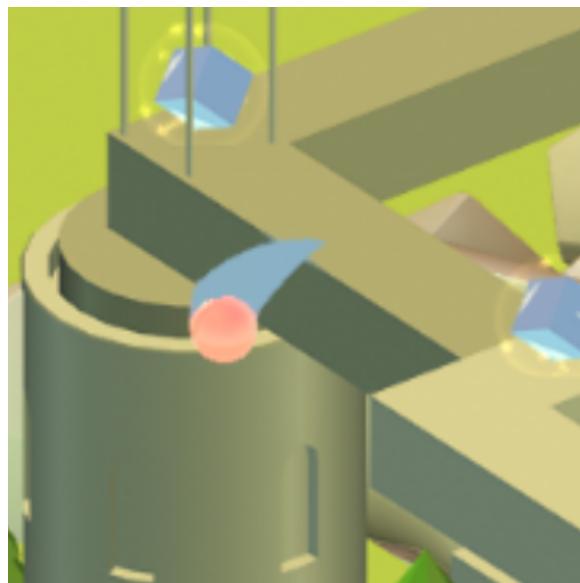
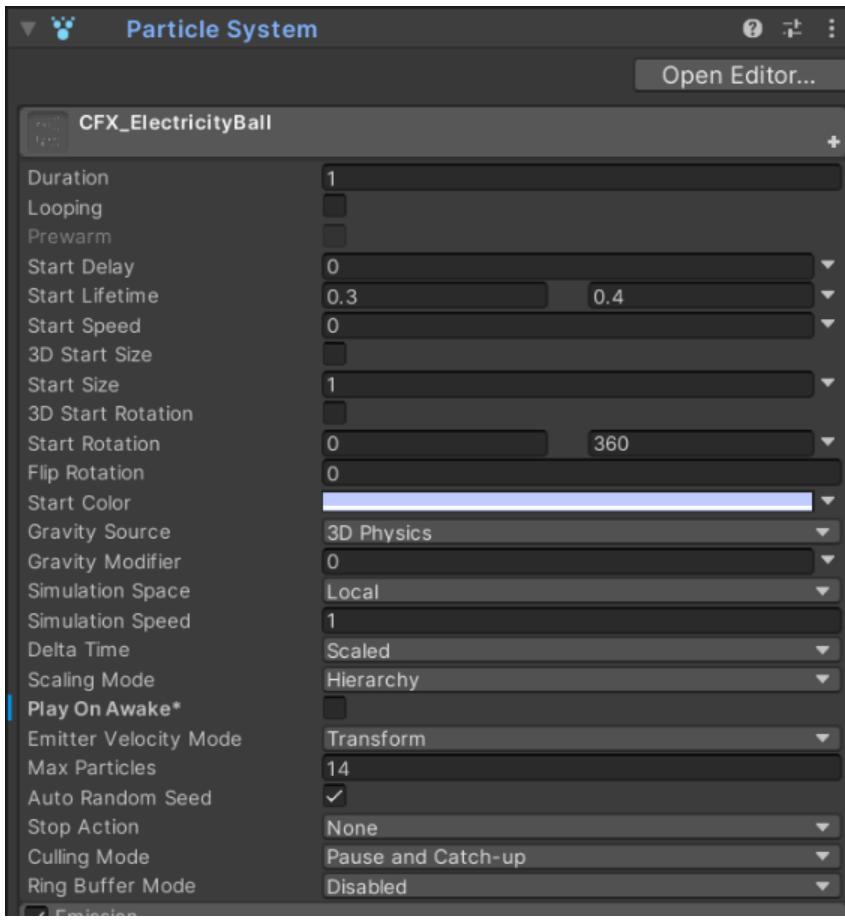
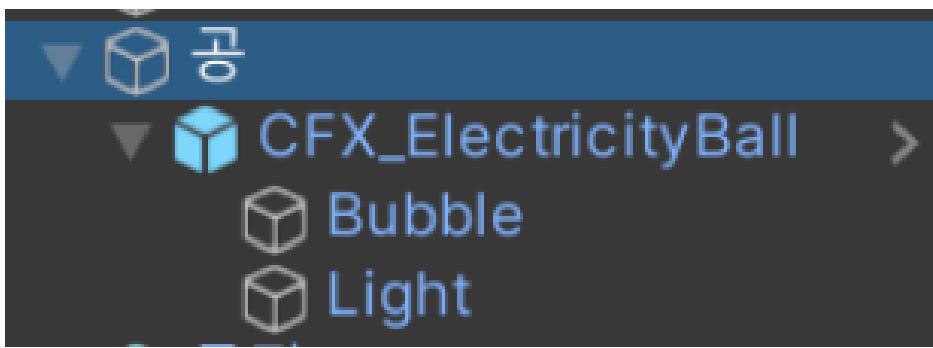
**Shadow**

- Script: Shadow
- Effect Color: [color swatch]
- Effect Distance: X 0.5 Y -1
- Use Graphic Alpha: ✓

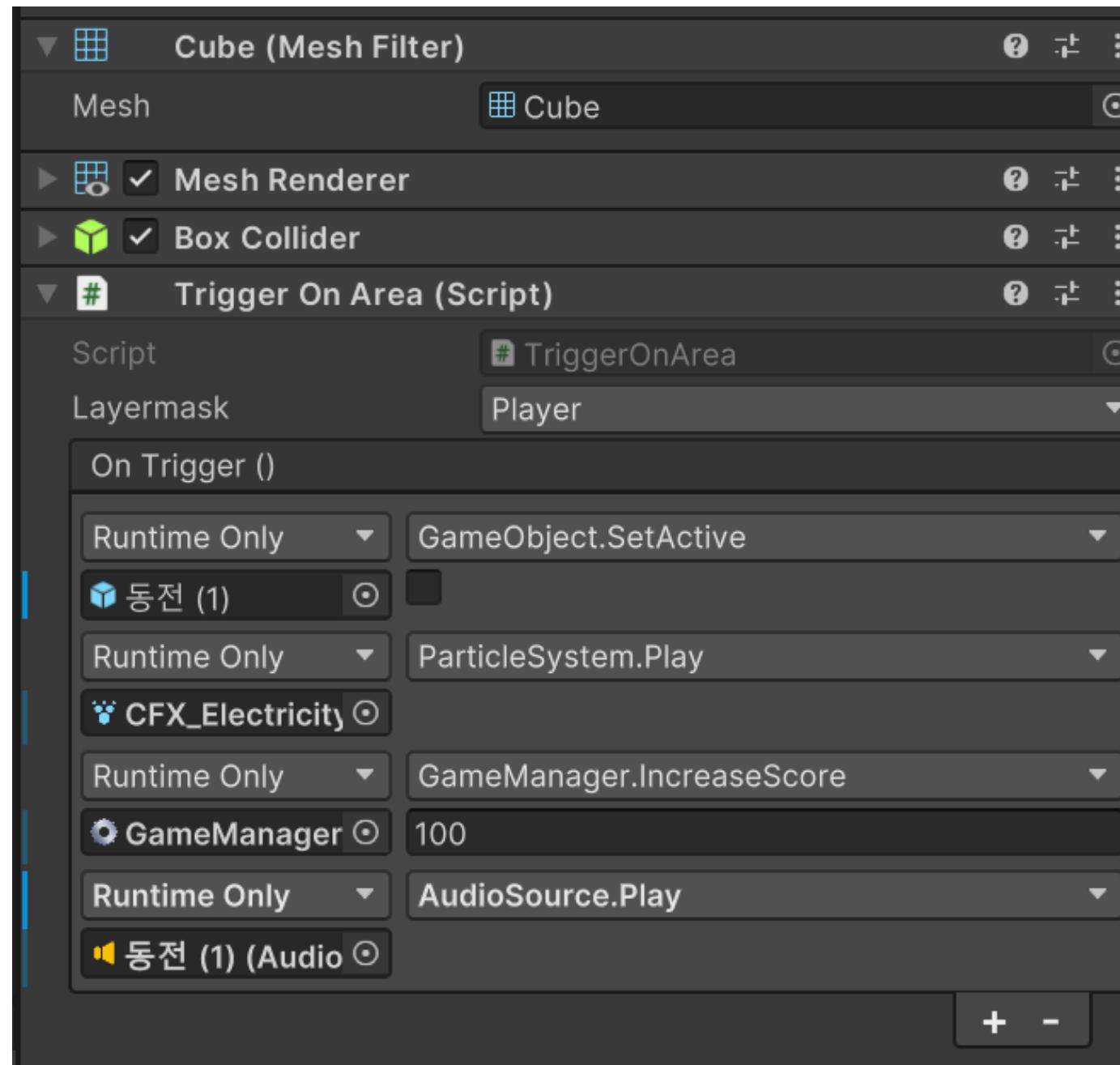
**Default UI Material (Material)**

- Shader: UI/Default

# Ball 설정 및 효과



# Coin 설정 및 효과



```
private AudioSource[] arrayAudio;

void Awake(){ StartGame(); }

void Start(){
    arrayAudio = GameObject.Find("GameManager").GetComponents<AudioSource>();
}

public void IncreaseScore(int score){
    _score = _score + score;
    DisplayScore();

    if (arrayAudio != null && arrayAudio.Length > 1){
        arrayAudio[1].Play();
    }
}
```

# FONT ASSET CREATOR

TextMesh Pro 폰트 어셋을 만들려면

1) 메뉴에서 선택:

창(Windows) > TextMesh Pro > Font Asset Creator를 선택

Font Asset Creator를 엽니다.

2) 소스 폰트 파일 선택:

Unity 폰트 어셋을 선택하여 TextMesh Pro 폰트 어셋으로 변환하  
려는 대상 파일을 선택

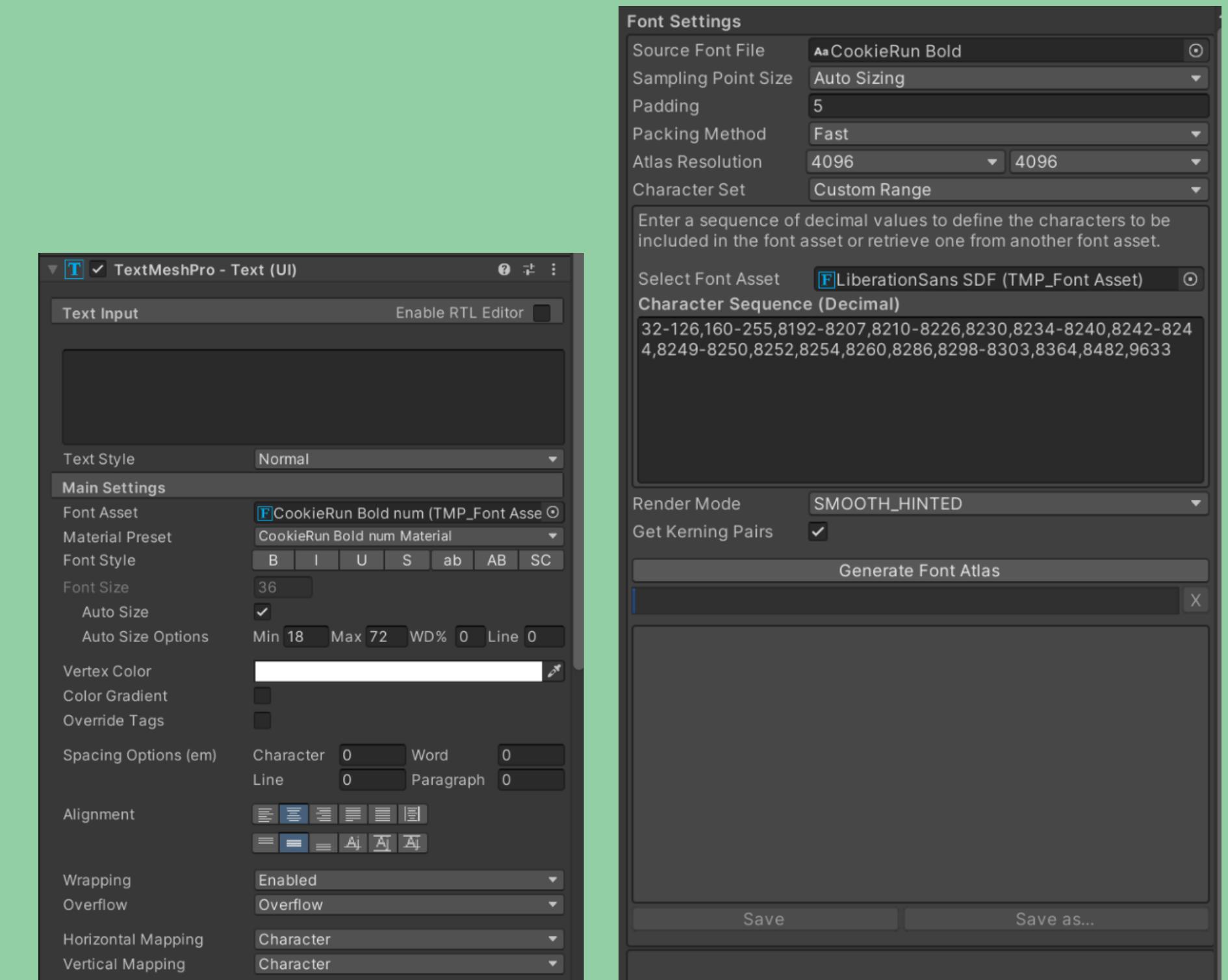
3) 폰트 설정 조정: 필요한 대로 폰트 설정을 조정한 다음,

"Generate Font Atlas"를 클릭하여 아틀라스 텍스처를 생성

4) 저장: "Save" 또는 "Save as..."를 클릭

5) Resources 폴더에 저장:

TextMesh Pro에서 액세스할 수 있도록 폰트 어셋을 Resources  
폴더에 저장해야 합니다.



# Post Process start

포스트 프로세싱은 기존에 렌더링된 씬에 렌더링 효과를 더하는 작업입니다. 포스트 프로세싱의 효과는 일반적으로 Scene 뷰에 따라 달라지거나, 최종 렌더링 결과물을 생성하기 전에 렌더링 되는 씬 위에 겹쳐서 표시됩니다. 이 기능은 기존 콘텐츠를 수정할 필요 없이 시각적인 효과를 즉시 구현하고 씬의 품질을 한층 개선할 수 있다는 뛰어난 장점이 있습니다.



# RESTART



```
public class GameManager : MonoBehaviour{  
  
    [SerializeField] private ConstantVelocityMovement _ball;  
    [SerializeField] private Transform _startPoint;  
    [SerializeField] private List<GameObject> _coins;  
    [SerializeField] private TextMeshProUGUI _scoreText;  
  
    private int _score = 0;  
    private AudioSource[] arrayAudio;  
  
    void Awake(){ StartGame(); }  
  
    void Start(){  
        arrayAudio = GameObject.Find("GameManager").GetComponents< AudioSource>();  
    }  
  
    public void IncreaseScore(int score){  
        _score = _score + score;  
        DisplayScore();  
  
        if (arrayAudio != null && arrayAudio.Length > 1){  
            arrayAudio[1].Play(); } }  
  
    public void ResetScore(){  
        _score = 0; DisplayScore(); }  
  
    private void DisplayScore(){  
        _scoreText.text = _score.ToString(); }  
  
    public void RestartGame(){ StartGame(); }  
  
    public void StartGame(){  
        _ball.Teleport(_startPoint.position);  
  
        foreach (GameObject coin in _coins){  
            coin.SetActive(true);  
        }  
  
        ResetScore(); }  
  
    public void EndGame(){}
}
```

# 업그레이드 포인트



upgrade 11++

## GAME MANAGER

1. serializeField 사용
2. Score 재구성
3. background sound 설정

## VISUAL and SOUND

1. font 변경 - font asset creator
2. asset sky background
3. ball upgrade particle 추가 및 tail 추가
4. coin upgrade: 습득 시 particle 및 효과음 추가
5. Post Process
6. Animation Curve

## PALY

1. Stage upgrade
2. intro, restart 재구성

# 감사합니다

질문이 있다면 말씀해주세요.

<https://junbastick.tistory.com/6>

<https://docs.unity.cn/kr/2021.3/Manual/animeditor-AnimationCurves.html>

<https://docs.unity3d.com/Packages/com.unity.textmeshpro@4.0/manual/FontAssetsCreator.html>